URL: http://www.elsevier.nl/locate/entcs/volume29.html 21 pages

A Bi-Categorical Axiomatisation of Concurrent Graph Rewriting ⋆

Fabio Gadducci [a] Reiko Heckel [b] Mercé Llabrés [c]

[a] *Un. of Edinburgh, Division of Informatics, Laboratory for Foundations of Computer Science,* fabio@dcs.ac.ed.uk

[b] *Un. of Paderborn, Department of Mathematics and Computer Science,* reiko@uni-paderborn.de

[c] *Un. of the Balearic Islands, Department of Mathematics and Computer Science,* merce@ipc4.uib.es

Abstract

In this paper the concurrent semantics of double-pushout (DPO) graph rewriting, which is classically defined in terms of shift-equivalence classes of graph derivations, is axiomatised via the construction of a free monoidal bi-category. In contrast to a previous attempt based on 2-categories, the use of bi-categories allows to define rewriting on *concrete* graphs. Thus, the problem of composition of isomorphism classes of rewriting sequences is avoided. Moreover, as a first step towards the recovery of the full expressive power of the formalism via a purely algebraic description, the concept of *disconnected rules* is introduced, i.e., rules whose interface graphs are made of disconnected nodes and edges only. It is proved that, under reasonable assumptions, rewriting via disconnected rules enjoys similar concurrency properties like in the classical approach.

1 Introduction

The *theory of graph transformation* [28] basically studies a variety of formalisms which extend the theories of formal languages and term rewriting, respectively, in order to deal with structures more general than strings and terms. In both of these "classical" formalisms there are two different ways of defining the entailment relation. For example, the *operational definition* of the rewrite relation $\Rightarrow_{\mathfrak{R}}$ for a term rewriting system $\mathfrak{R}$ states that a rewrite rule

⋆ Research partially supported by the British EPSRC grant R29375 through the University of Edinburgh; by the ESPRIT BR Working Group APPLIGRAPH through the University of Paderborn; and by the Spanish DGES project PB96-0191-CO2 through the University of the Balearic Islands.

$l \to r \in \mathfrak{R}$ is applicable to a term t if an instance of l occurs as a sub-term in t. Then, this sub-term may be removed and replaced by a corresponding instance of r, leading to a derived term s. Equivalently, an *inductive definition* may be given where the rewrite relation is obtained as the smallest relation which contains $\mathfrak{R}$ and is closed under substitution and context. While the operational definition is clearly more intuitive, the inductive one plays an important role in the theory since it allows for definitions and proofs by structural induction. From a categorical view-point, such inductive definitions have been given for various formalisms via the construction of free categories equipped with an orthogonal (algebraic or categorical) structure [3,7,8,10,16,25–27,29,30]. Often such categorical models of rewriting do not only axiomatise the rewrite relation but impose an equivalence on rewriting sequences which captures the basic concurrency properties of the system.

In the *double-pushout* (DPO) approach to graph transformation [11,15] the operational definition is by far more popular. Inductive definitions of DPO graph transformation have been given but they are not as well accepted as, e.g., in the theory of term rewriting. One reason may be that, unlike for strings and terms, there is no straightforward inductive definition of graphs. Rather, each possible interpretation suggests a different choice of the basic operations, see for example [1,10,16,19] for different formulations of the DPO approach. Another reason might be that, except for the last and most recent one, none of the above formulations models faithfully the concurrent semantics of DPO graph rewriting based on the so-called *shift-equivalence* of derivations which captures the abstraction from the execution order of independent steps [11,23].

The reason for this failure is two-fold. Some approaches [1,10,16] define rewriting on (partly) abstract graphs without providing appropriate means for the composition of isomorphism classes of arrows. As a consequence, many derivations which are not shift-equivalent are identified. From the operational point of view, this problem was recognised and solved in [6,11] by the concept of *standard isomorphisms*, i.e., a chosen family of isomorphisms closed under composition and identities which are used to compose isomorphism classes of rewriting sequences. An axiomatic description of this solution, however, has not been provided yet. Other approaches, like [1] and [16], are only applicable in very restricted cases: They only allow graph rewrite productions $L \longleftarrow K \to R$ with discrete interface graphs K (i.e., without edges). This makes no harm if one is only interested in the generated rewrite relation since the preservation of an edge can be modelled via its deletion and re-generation. However, by "making discrete" the interface of a production, the set of possible parallel derivations is reduced since items that are shared in a parallel application have to be preserved by all the applied productions. (This is very similar to data base transactions where *read locks* may be held by several transactions at the same time while *write locks* are exclusive.)

The presentation in this paper refines the approach of [16] where the rewrite relation of a graph rewrite system has been characterised via the construction

of a free monoidal 2-category. The conceptual idea of [16] is to consider graphs as distributed states consisting of local components connected through interfaces. The distributed structure is made explicit by representing a graph as an *arrow* of a category of co-spans in the usual category of graphs and graph homomorphisms: Each arrow represents a local component, and its source and target objects are the interfaces through which it is connected to other components. Then, arrow composition, defined via pushouts, represents the composition of two components over a common interface.

Since pushouts are only associative up-to isomorphism, the associativity law of horizontal composition in a 2-category of co-spans implies that all isomorphic co-spans with the same source and target are actually equal. This leads to the above-mentioned problems with the composition of rewrite steps. For this reason, in this paper the associativity of horizontal composition is dropped and replaced by a vertical isomorphism, that is, 2-categories are replaced by bi-categories. Moreover, the restriction of [16] to discrete interface graphs (i.e., sets of interface vertices) is relaxed by allowing *disconnected* interfaces (that is, isolated nodes and edges). This solves the above-mentioned problems with respect to concurrency properties.

But why do we not simply allow for arbitrary graphs as interfaces? Again there are two reasons, a technical and a conceptual one. Conceptually, interfaces in distributed systems are usually much simpler than the component themselves. In fact, it is one of the main principles of system design to minimise the relations between different components or modules and to maximise the internal connections instead. Technically, disconnected graphs are simpler because they can be freely generated by a monoidal operation (disjoint union) from single nodes and edges, which is clearly not possible for arbitrary graphs.

Different restrictions of the set of admissible productions have been studied in the literature on graph rewriting in order to show that the resulting class of graph derivations enjoys interesting concurrency properties. In particular, it is almost customary to consider only *injective* productions, as e.g. in the survey article [11]. However, since productions obtained by disconnecting the interfaces of injective productions are typically non-injective, in this paper no general assumptions can be made about the injectivity of production morphisms. Thus, we investigate the expressive power of graph rewriting via non-injective productions showing that (suitable sub-classes of) productions with disconnected interfaces actually satisfy the desired concurrency properties.

After recalling in Section 2 some simple properties about *parallelism*, and introducing the original notion of *as-correspondence*, this comparison is done in Section 3. We show that for a system $\mathcal{G}$ satisfying the usual restriction to injective productions, a "disconnected" system $\delta(\mathcal{G})$ can be built which induces the same rewrite relation. Moreover, we establish a one-to-one correspondence between the derivations in $\mathcal{G}$ and a suitable subclass of the derivations in $\delta(\mathcal{G})$ such that two derivations in $\mathcal{G}$ are shift-equivalent if and only if this is true

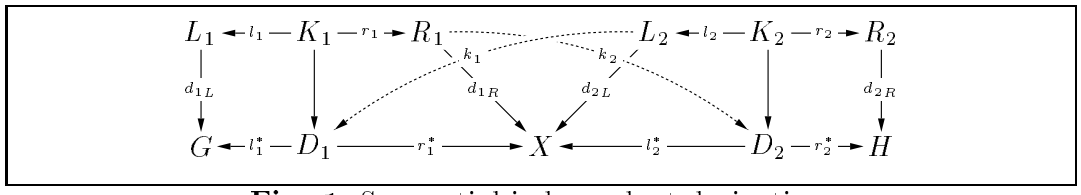

Fig. 1: Sequential independent derivation.

for the corresponding derivations in $\delta(\mathcal{G})$.

In Section 4 we introduce the notion of *dgs-monoidal bi-category*, then presenting our category of co-spans of graphs as dgs-monoidal bi-category. This work is just preparatory for Section 5, where we provide a bi-categorical axiomatisation of shift-equivalence for graph derivations in $\delta(\mathcal{G})$.

2 Concurrent graph transformation *via* analysis-synthesis

Appendix A.1 recalls the basic definitions regarding the double-pushout (DPO) approach to graph transformation in the case of general productions, i.e., without the usual assumption that one or both components of a production span are injective. In this section we introduce, in the generalised setting, concurrent derivations in a GTS $\mathcal{G}$ as equivalence classes of concrete derivations up-to *shift-equivalence*. The idea is to identify derivations which differ only for the scheduling of independent steps. Such equivalence classes are similar to the derivation traces in [11] but for the use of concrete graphs (instead of isomorphism classes). Thus we avoid the problems of standard isomorphisms, and this motivates our use of bi- instead of 2-categories (as in [16]). First, we introduce the basic notions of independence and parallelism of DPO graph transformation. Then, we define shift-equivalence based on a new (and more compact) presentation of the correspondence between sequential and parallel derivations induced by the classical parallelism theorem.

Definition 2.1 (sequential independence) *Let* $G \overset{p_1/d_1}{\Longrightarrow} X \overset{p_2/d_2}{\Longrightarrow} H$ *be a two-steps derivation, as in Figure 1. We call it* sequential independent *if there exist two graph morphisms* $R_1 \overset{k_2}{\longrightarrow} D_2$ *and* $L_2 \overset{k_1}{\longrightarrow} D_1$ *such that* $l_2^* \circ k_2 = d_{1R}$ *and* $r_1^* \circ k_1 = d_{2L}$. □

Intuitively, two consecutive steps $G \overset{p_1/d_1}{\Longrightarrow} X \overset{p_2/d_2}{\Longrightarrow} H$ are sequentially independent if they may be swapped, i.e., if p_2 can be applied to G, and p_1 to the resulting graph. Under suitable conditions, such a derivation can be simulated by the *parallel* application of the two underlying productions.

Definition 2.2 (parallel production and derivation) *Let* $p_1 : s_1$ *and* $p_2 : s_2$ *be productions. The associated* parallel production $p_1 \oplus p_2 : s_1 + s_2$ *is denoted by the (component-wise) co-product of the spans* s_1 *and* s_2. *A* direct parallel derivation $G \overset{p_1 \oplus p_2/d}{\Longrightarrow} H$ *is then a direct derivation using a parallel production. A* parallel derivation *is a finite sequence of direct parallel derivations.*

The set of all parallel derivations in a GTS $\mathcal{G}$ *is denoted by* $D(\mathcal{G})$. □

We implicitly assume in the above definition that we can recursively build parallel productions out of already parallel ones. Unless otherwise noted, in the following derivation stands for *parallel* derivation.

The parallelism theorem states that a direct parallel derivation can be sequentialised into sequential independent applications of the component productions. And vice versa, that two consecutive steps can be put in parallel if they are sequential independent.

Theorem 2.3 (parallelism) *Let* p_1, p_2 *be (possibly parallel) injective graph productions. Then, the following statements are equivalent:*

(i) *There is a direct parallel derivation* $\rho = G \overset{p_1 \oplus p_2/d}{\Longrightarrow} H$.

(ii) *There is a sequentially independent derivation* $\sigma = G \overset{p_1/d_1}{\Longrightarrow} X \overset{p_2/d_2}{\Longrightarrow} H$. □

In the classical formulation of this theorem [13], the correspondence between the derivations in (i) and (ii) is established by an *analysis* construction (from ρ to σ) and a *synthesis* construction (from σ to ρ). Both constructions produce, as intermediate result, two DPO steps $G_i \overset{p_i/c_i}{\Longrightarrow} H_i$ for $i = 1, 2$ with the same context graph D like the parallel derivation ρ. Then, two different schedulings are employed leading, respectively, to the parallel and sequential derivation of 1. and 2. above. The *parallel scheduling* $G \overset{p_1 \oplus p_2/d}{\Longrightarrow} H$ is constructed by gluing the given and derived graphs of the two steps over the common interface D leading to graphs G and H, respectively. The *sequential scheduling* $G \overset{p_1/d_1}{\Longrightarrow} X \overset{p_2/d_2}{\Longrightarrow} H$ is obtained by forming, in addition, the intermediate graph X as gluing of H_1 and G_2 and composing the embedded steps sequentially.

Thus, we may consider instead as primitive the correspondence established by the analysis and synthesis construction, and we obtain a descriptive way for relating the parallel and sequential schedulings of *all pairs* of DPO steps ρ_1 and ρ_2 over the same context graph D.

Definition 2.4 (as-correspondence) *Let* $\rho_i = G_i \overset{p_i/c_i}{\Longrightarrow} H_i$ *for* $i = 1, 2$ *be direct derivations with the same context graph* D. *A derivation* $\rho = G \overset{p/d}{\Longrightarrow} H$ *is called a* parallel scheduling *of* ρ_1 *and* ρ_2 *if in the left diagram of Figure 2 (where the solid part represents the given steps) graphs* G *and* H *are constructed by pushouts (1) and (2), respectively;* $p = p_1 \oplus p_2$ *and* ρ *is realized by the DPO diagam in Figure 3.*

The derivations $\sigma = G \overset{p_1/d_1}{\Longrightarrow} X \overset{p_2/d_2}{\Longrightarrow} H$ *and* $\sigma' = G \overset{p_2/d_1'}{\Longrightarrow} Y \overset{p_1/d_1'}{\Longrightarrow} H$ *are called* sequential schedulings *of* ρ_1 *and* ρ_2 *if* σ *is constructed as in the right diagram of Figure 2 where graphs* G, H *and* X *are obtained by pushouts (1), (2) and (3), respectively;* d_1 *and* d_2 *are the DPO diagrams from* $L_1 \leftarrow K_1 \rightarrow R_1$ *to* $G \leftarrow G_2 \rightarrow X$ *and from* $L_2 \leftarrow K_2 \rightarrow R_2$ *to* $X \leftarrow H_1 \rightarrow H$, *and symmetrically*

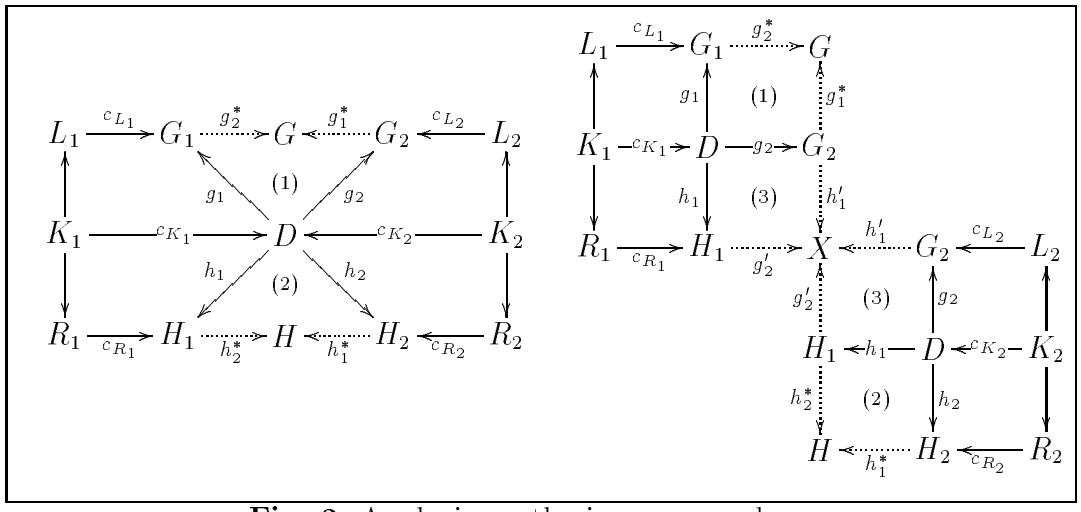

Fig. 2: Analysis-synthesis correspondence.

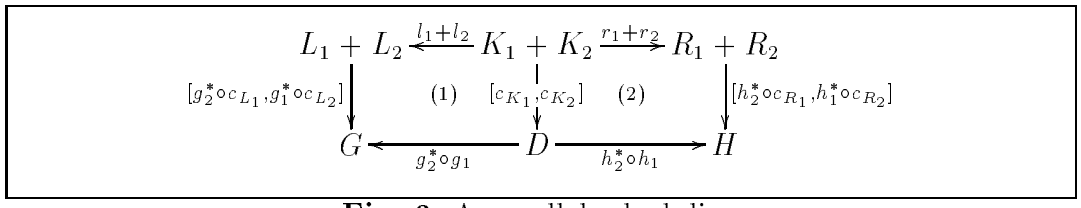

Fig. 3: A parallel scheduling

for σ'.

Two derivations ρ and σ are in analysis-synthesis correspondence*, briefly* as-correspondence*, if there are steps ρ_i for $i = 1, 2$ such that ρ is a parallel scheduling and σ is a sequential scheduling of ρ_1 and ρ_2.*

Finally, the shift-equivalence $\equiv_{sh} \subseteq D(\mathcal{G}) \times D(\mathcal{G})$ *is the least equivalence relation on derivations containing the as-correspondence, and which is closed under sequential composition of derivations.* □

Denoting the parallel scheduling by $\rho_1|\rho_2$ and the two sequential schedulings by $\rho_1;\rho_2$ and $\rho_2;\rho_1$, respectively, the above definition could be summarised by the equation $\rho_1;\rho_2 \equiv_{sh} \rho_1|\rho_2 \equiv_{sh} \rho_2;\rho_1$. Notice, however, that ";" and "|" are only defined up to isomorphism by the colimit constructions above, i.e., they are not operations in the algebraic sense.

3 Concurrency for disconnected productions

In this section we introduce *disconnected productions*, showing that they preserve the same degree of concurrency of injective ones.

Definition 3.1 (disconnected graphs and productions) *Let $\langle E, N, s, t\rangle$ be a graph. A node $n \in N$ is* isolated *if $n \notin s(E) \cup t(E)$. The graph is* discrete *if $E = \emptyset$; and it is is* disconnected *if s, t are jointly injective, i.e., both are*

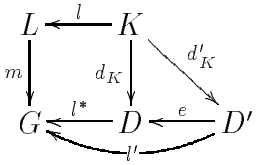

Fig. 4: Decomposition for pushout complement

injective and $s(E) \cap t(E) = \emptyset$.[1]

A production $p : (L \xleftarrow{l} K \xrightarrow{r} R)$ *is* disconnected (discrete) *if so is the interface graph* K. *A graph transformation system* $\mathcal{G}$ *is* disconnected (discrete) *if all its productions are so.* □

For productions with injective left-hand side it is well-known [11] that the pushout complements are unique up-to isomorphism. In general, this is not the case. However, we may obtain a chosen isomorphism class using *natural pushout complement* [14], characterised by the following universal property.

Definition 3.2 (natural pushout complement) *Let* l *and* m *be morphisms as in Fig. 4. A pushout complement* $\langle D, l^*, d_K \rangle$ *is called* natural *if for any other pushout complement* $\langle D', l', d'_K \rangle$ *of* l *and* m *there exists a surjective morphism* $e : D' \to D$ *making the diagram in Fig. 4 commute.*

The set of all derivations in a GTS $\mathcal{G}$ via *natural pushout complements is denoted by* $D_n(\mathcal{G})$. □

For a given rule p and match m satisfying the gluing conditions, a natural pushout complement can be always built, simply formalising the operational intuition about derivation that we sketched below Definition A.3. In fact, we just need to consider the obvious arrow $d_k : K \to G - m(L - l(K))$. It is easy to show that the morphism l^* is always injective, and that all natural pushout complements are isomorphic: We then recover the well-known uniqueness result for injective productions, proving that all DPO derivations involving injective productions are always natural.

The following definition provides a canonical way of "disconnecting" a graph: Applying this construction to the interface graph of a production yields the corresponding disconnected production.

Definition 3.3 (minimal disconnected graphs and productions) *Let* G *be a graph. We denote by* $\delta(G)$ *the* minimal disconnected graph *underlying* G, *that is, the pair* $\langle \bar{G}, \delta_G \rangle$, *where* $\bar{G}$ *is the graph freely generated by the set of edges* E_G *and the set of isolated nodes in* N_G, *and* $\delta_G : \bar{G} \to G$ *is the obvious surjective graph morphism.*

Let $p : s$ *be a production. The associated disconnected production is given by* $\delta(p) : (L \xleftarrow{l \circ \delta_K} \bar{K} \xrightarrow{r \circ \delta_K} R)$. □

[1] In other words, a node is isolated if it has nor incoming neither outcoming edges; a graph is discrete if all its nodes are isolated; and it is disconnected if it is "freely" generated by a set of edges and a set of isolated nodes.

Since the disconnected graph $\bar{K}$ is uniquely (albeit informally) determined, the previous definition actually describes a function $\delta : \mathfrak{R} \to \mathfrak{R}_d$ from the set of all productions to the set of disconnected productions. This function allows us to associate to each GTS $\mathcal{G}$ a disconnected GTS $\delta(\mathcal{G}) = \{\delta(p : s) \mid p : s \in \mathcal{G}\}$. The following proposition shows that both systems induce the same rewrite relation over graphs.

Proposition 3.4 (disconnected rewrites) *Let $\mathcal{G}$ be a* GTS*, and let $\delta(\mathcal{G})$ be the associated disconnected* GTS*. Then there exists a direct derivation $G \Longrightarrow H$ in $\mathcal{G}$ if and only if there exists a direct derivation $G \Longrightarrow H$ in $\delta(\mathcal{G})$.* □

Please note that the proposition above does *not* imply that there is a one-to-one correspondence between the class of direct derivations of $\mathcal{G}$ and of $\delta(\mathcal{G})$: The latter is larger since more interface graphs are allowed. Then, in order to compare the concurrency properties of a GTS $\mathcal{G}$ and its associated disconnected GTS $\delta(\mathcal{G})$, we restrict $\mathcal{G}$ to injective productions and consider derivations in $\delta(\mathcal{G})$ via natural pushout complements only, denoting such a class as $D_n(\delta(\mathcal{G}))$.

Proposition 3.5 (correspondence with disconnected derivations) *Let $\mathcal{G}$ be an injective* GTS *and $\delta(\mathcal{G})$ its associated disconnected one. Then, there exists a bijective function $\delta : D(\mathcal{G}) \to D_n(\delta(\mathcal{G}))$ which associates to each direct derivation $G \overset{p/d}{\Longrightarrow} H$ in $D(\mathcal{G})$ (like in Figure A.1) a direct derivation $G \overset{\delta(p)/\delta(d)}{\Longrightarrow} H$ with $\delta(d) = \langle d_L, d_K \circ \delta_K, d_R \rangle$.*

Proof (Sketch). The surjectivity of $\delta_K : \bar{K} \to K$ (the morphism obtained by "disconnecting" p's interface graph according to Definition 3.3) ensures that $\delta(p/d)$ is indeed a DPO step *via* natural pushout complement. The one-to-one correspondence is proved by epi-mono factorisation, which is used to reconstruct the original production and direct derivation. □

The function δ extends to derivations in the obvious way, providing a one-to-one correspondence $\delta : D(\mathcal{G}) \to D_n(\delta(\mathcal{G}))$ between derivations in $\mathcal{G}$ and derivations via natural pushout complements in $\delta(\mathcal{G})$. Next result shows that it preserves the concurrency properties of derivations, that is, sequential independence and shift-equivalence.

Theorem 3.6 (concurrency for disconnected productions) *Let $\mathcal{G}$ be an injective* GTS*.*

(i) *Let p_1 and p_2 be two (possibly parallel) productions in $\mathcal{G}$. Then, $\rho = G \overset{p_1/d_1}{\Longrightarrow} H_1 \overset{p_2/d_2}{\Longrightarrow} H_2 \in D(\mathcal{G})$ is a sequential independent derivation if and only if $\delta(\rho)$ is sequential independent.*

(ii) *Let ρ and σ be two derivations in $D(\mathcal{G})$. Then, $\rho \equiv_{sh} \sigma$ if and only if $\delta(\rho) \equiv_{sh} \delta(\sigma)$.*

Proof (Sketch). The first statement above is an obvious consequence of Definition 2.1 and the definition of δ in Proposition 3.5: The independence property

does not involve the interface graph and its outgoing morphisms, and all other components are left untouched by δ. The second statement follows by showing that two derivations are in as-correspondence if and only if this is true for their images under δ. To this aim we again exploit the surjectivity of the "disconnection arrows" δ_{K_i} as well as their compatibility with colimits. □

We remark that the above theorem is the main rational behind the introduction of disconnected productions. In fact, it is well-known since e.g. [1] that discrete GTS's preserve the generative power of general GTS's, but not their degree of concurrency. Disconnected GTS's are then better-behaved, still maintaining a relatively simple interface, and this is relevant from a practical point of view, as argued in the introduction.

4 On some structures for bi-categories

Appendix A.2 recalls the basic definitions regarding *monoidal bi-categories*: Most of them are standard, and can be found in classical references [2] (even if our presentation follows closely the recent survey [24]), except for monoidality, for which we refer to [17]. In Section 4.1 we first introduce *pseudo monoids* [12], then presenting our personal addition to the bi-categorical folklore, namely, *dgs-monoidal* bi-categories, and spelling out their relationship with *cartesian* bi-categories [5]. The notion of monoidal bi-categories is the most relevant for the main results of Section 5, and the reader could then skip Section 4.1 at a first reading, except for a few notational conventions. The latters are exploited in Section 4.2, presenting some easy results on *bi-categories of co-spans*.

4.1 Cartesian and dgs-monoidal bi-categories

Next definition is borrowed from Section 3 of [12], and (slightly) generalised in order to deal with monoidal bi-categories, instead of just Gray monoids.

Definition 4.1 (pseudo monoids) *Let a be an object of a monoidal bi-category $\underline{\mathbf{C}}$. A* pseudo monoid *for a is a five-tuple $\langle \Delta_a, ?_a, \alpha'', \lambda'', \rho'' \rangle$ such that $\Delta_a : a \otimes a \to a$ and $?_a : e \to a$ are arrows of $\underline{\mathbf{C}}$; and $\alpha''_a : \alpha'_{a,a,a} * ((\Delta_a \otimes id_a) * \Delta_a) \Rightarrow (id_a \otimes \Delta_a) * \Delta_a$, $\rho''_a : (?_a \otimes id_a) * \Delta_a \Rightarrow \rho'_a$ and $\lambda''_a : (id_a \otimes ?_a) * \Delta_a \Rightarrow \lambda'_a$ are invertible cells, satisfying the axioms*[2]

- $((\alpha'' \otimes id_a) * \Delta_a) \cdot ((id_a \otimes \Delta_a \otimes id_a) * \alpha'') \cdot ((id_a \otimes \alpha'') * \Delta_a)$
  $= ((\Delta_a \otimes id_a \otimes id_a) * \alpha'') \cdot (\xi * \Delta) \cdot (id_a \otimes id_a \otimes \Delta_a) * \alpha''$, *and*
- $((id_a \otimes \Delta_a) \otimes id_a) * \alpha'' = (\rho''_a \otimes id_a) * \Delta_a$,

for ξ the unique arrow induced by the monoidal structure.

[2] For the sake of readability, and since they do not play a relevant part for this paper, we spelled out incompletely those coherence axioms, skipping the isomorphism cells induced by the bi-categorical structure.

A pseudo co-monoid *for a is a pseudo monoid $\langle \nabla, !, \alpha'', \lambda'', \rho'' \rangle$ for a in the dual monoidal bi-category $\underline{\mathbf{C}}^{op}$, obtained reversing the arrows of $\underline{\mathbf{C}}$.* □

Bi-categories equipped with suitable pseudo (co-)monoids enjoy rather strong properties. As an example, it is possible to prove, generalising Proposition 4 of [12], that for a, b objects of a monoidal bi-category $\underline{\mathbf{C}}$ equipped with a pseudo co-monoid and a pseudo monoid, respectively $\underline{\mathbf{C}}[a, b]$ can be equipped with a monoidal category.

Definition 4.2 (dgs-monoidal bi-categories) *Let a be an object of a monoidal bi-category $\underline{\mathbf{C}}$. We call it* discrete *[5] if it is equipped with a pseudo monoid and a pseudo co-monoid, and an invertible cell $\Delta_a * \nabla_a \Rightarrow (id_a \otimes \nabla_a) * \alpha'_{a,a,a} * (\Delta_a \otimes id_a)$; and* functional *if it is also equipped with an invertible cell $\nabla_a * \Delta_a \Rightarrow id_a$.*

We call a bi-category dgs-monoidal *if each object is discrete and functional.* □

We introduced *dgs-monoidal categories* in [16] in order to model a suitable category of (abstract) graphs.[3]. Related presentations surfaced quite frequently in recent years. In particular, a similar structure is used for the description of bi-categories of (co-)spans already in [4,5], and that presentation forms the basis for the categorical description of *circuits* [20,21]. We close the section trying to make explicit such a relationship.

Definition 4.3 (adjoints and co-cartesian bi-categories) *Let $f : a \to b$, $g : b \to a$ be arrows of a bi-category An* adjunction *[22] $\eta, \epsilon : f \lhd g$ consists of cells $\eta : id_a \Rightarrow f * g$ and $\epsilon : g * f \Rightarrow id_b$ satisfying the axioms*

- $(g * \eta) \cdot \alpha_{g,f,g} \cdot (\epsilon * g) = \lambda_g \cdot \rho_g^{-1}$, *and*
- $(\eta * f) \cdot \alpha_{f,g,f} \cdot (f * \epsilon) = \rho_f \cdot \lambda_f^{-1}$.

We say that f is a left-adjoint of g, or equivalently, that g is a right-adjoint of f; thus, η and ϵ are the unit and the co-unit of the adjunction, respectively. An adjunction is a reflection *if η is an isomorphism cell; it is a* co-reflection *if ϵ is an isomorphism cell.*

A co-cartesian *bi-category [5] is a monoidal bi-category such that each object a is equipped with a pseudo monoid and a pseudo co-monoid, and arrows ∇_a and $!_a$ have left-adjoints Δ_a and $?_a$, respectively. A* bi-category of relations *is a co-cartesian bi-category such that each object is discrete.* □

Thus, our dgs-monoidal bi-categories actually lack only suitable cells $\eta_a : id_{a \otimes a} \Rightarrow \Delta_a * \nabla_a$, $\eta'_a : id_e \Rightarrow ?_a * !_a$ and $\epsilon'_a : !_a * ?_a \Rightarrow id_a$ to be bi-categories of relations, where the adjoint $\Delta_a \lhd \nabla_a$ is a co-reflection (that is, the co-unit

[3] The structure we used here is weaker, since we are not assuming the existence of a *symmetry* for the monoidal bi-category. Such a restricted version suffices however our introductory purposes.

$\epsilon_a : \nabla_a * \Delta_a \Rightarrow id_a$ is an isomorphism cell).[4]

4.2 Some results on bi-categories of spans

The paradigmatic example of dgs-monoidal (as well as co-cartesian) bi-categories are bi-categories of *co-spans*.

Definition 4.4 (bi-categories of co-spans) *Let* $\mathbf{C}$ *be a category with* chosen *binary co-products, pushouts and initial object, and let us consider the seven-tuple* $\langle Ob_{\mathbf{C}}, CoSpan(\mathbf{C}), *, id, \alpha, \rho, \lambda\rangle$*, where* $Ob_{\mathbf{C}}$ *is the set of objects of* $\mathbf{C}$*; the arrows of* $CoSpan(\mathbf{C})[a,b]$ *are the triples* $\langle f, c, g\rangle$ *for* $f : a \to c$ *and* $g : b \to c$ *arrows in* $\mathbf{C}$*; the cells* $l : \langle f, c, g\rangle \Rightarrow \langle h, d, i\rangle$ *are those arrows* $l : c \to d$ *in* $\mathbf{C}$ *making the diagrams commute;* $id_a = \langle id_a, a, id_a\rangle$*; arrow (i.e., co-span) composition is defined by the chosen pushouts, inducing the cells* α*,* ρ *and* λ *by the universal property.* □

Proposition 4.5 *Let* $\mathbf{C}$ *be a category satisfying the conditions in Definition 4.4. Then,* $CoSpan(\mathbf{C})$ *is a bi-category of relations.* □

This proposition can be considered categorical folklore. The monoidal structure is obtained from the chosen co-products. The pseudo monoidal structure is given via the injections and mediating morphisms by choosing $\Delta_a = \langle [id_a, id_a], a, id_a\rangle$ and $?_a = \langle [a], a, id_a\rangle$, for $[id_a, id_a]$ and $[a]$ the co-pairing and initial arrow in $\mathbf{C}$, respectively.

Next proposition is also obtained by an explicit lifting of the structure in $\mathbf{C}$.

Proposition 4.6 *Let* $\mathbf{C}$ *be a category satisfying the conditions in Definition 4.4, and let* $CoSpan^{iso}(\mathbf{C})$ *be the bi-category included in* $CoSpan(\mathbf{C})$*, with the restriction that all the cells are actually isomorphisms in* $\mathbf{C}$*. Then,* $CoSpan^{iso}(\mathbf{C})$ *is a dgs-monoidal bi-category.* □

A bi-category of co-spans of graphs restricted to isomorphism cells shall be our environment category for the generation of graph derivations. The restriction to iso cells is necessary since cells are meant to represent rewrite steps which have to be explicitly specified by productions.

5 From DPO-rewrites to bi-categories

We already mentioned how most of the categorical descriptions of production-based systems [7,8,16,18,20,25,27,29,30] simulate computations *via* cells. To a certain extent, they all share the same view, representing such a system as a *computad* [31], namely, a category (in our case, a discrete bi-category) augmented with a graph structure over hom-sets (informally, a set S of cells

[4] Actually, in [5] the authors assume $\underline{\mathbf{C}}$ to be posetal, thus collapsing it to a 2-category. In fact, this is why they denote such bi-categories as "cartesian": In their case, the given structure is unique up-to isomorphism.

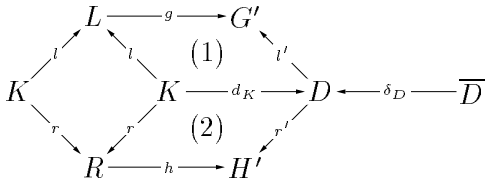

Fig. 5: The mapping of a direct derivation

not closed under any composition). In general terms, the states are the arrows of the computad, and the productions are its cells. Like a category can be freely generated from a graph, so a bi-category can be freely generated from a computad, closing the set of cells under all relevant operations.

Definition 5.1 (bi-computads) *A* bi-computad $\underline{\mathbf{C}}_S$ *is a pair* $\langle \underline{\mathbf{C}}, S \rangle$*, where* $\underline{\mathbf{C}}$ *is a monoidal bi-category and* S *is a set of "cells" over the hom-sets of* $\mathbf{C}$*. A* bi-computad morphism $\langle F, h \rangle : \underline{\mathbf{C}}_S \to \underline{\mathbf{D}}_T$ *is a pair such that* $F : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$ *is a monoidal morphism, and* $h : S \to T$ *is a function, preserving source and target of the cells in the expected way. Computads and their morphisms form a category, denoted* **BiComp**. □

Proposition 5.2 (free bi-categories) *Let* V_b : **BiCat** $\to$ **BiComp** *be the forgetful functor mapping a monoidal bi-category to the underlying computad: It admits a left adjoint* F_b : **BiComp** $\to$ **BiCat**. □

Let us see now how to derive a bi-computad from a graph transformation system $\mathcal{G}$. First note that, since **Graph** satisfies the condition of Definition 4.4, the bi-categories $CoSpan(\mathbf{Graph})$ and $CoSpan^{iso}(\mathbf{Graph})$ are well-defined.

Definition 5.3 *Let* $\mathcal{G}$ *be a disconnected* GTS *and let* $CoSpan^{\oplus,iso}(\mathbf{Graph})$ *denote the bi-category fully included in* $CoSpan^{iso}(\mathbf{Graph})$*, with disconnected graphs as objects. The associated computad* $\mathbf{C}_{\mathcal{G}}$ *has* $CoSpan^{\oplus,iso}(\mathbf{Graph})$ *as underlying bi-category, and a cell* $[p] : \langle l, L, l \rangle \Rightarrow \langle r, R, r \rangle$ *for each production* $p : (L \xleftarrow{l} K \xrightarrow{r} R) \in \mathcal{G}$. □

Cells representing derivations are generated by imposing on these *production cells* the operations of the monoidal bi-category.

Definition 5.4 (mapping direct derivations) *Let* $\mathcal{G}$ *be a disconnected* GTS*, and let* $G \overset{p/d}{\Longrightarrow} H$ *be a direct derivation as shown in Figure A.1.*

- *If* p *is an elementary (i.e., non-parallel) production in* $\mathcal{G}$*, then the cell* $[p/d]$ *associated to* $G \overset{p/d}{\Longrightarrow} H$ *is given by* $[p] * \langle d_K, D, \delta_D \rangle$.
- *If* $p = p_1 \oplus p_2$ *is a parallel production, then the cell associated to* $G \overset{p/d}{\Longrightarrow} H$ *is given by* $[p/d] = ([p_1/d \circ \iota_1] \otimes [p_2/d \circ \iota_2]) * \Delta_{\overline{D}}$ *for* $\iota_i : K_i \to K_1 + K_2$*,* $i = 1, 2$*, the injection morphisms for the coproduct.* □

The cell $[p/\langle g, d_K, h \rangle] = [p] * \langle d_K, D, \delta_D \rangle$, for p production in $\mathcal{G}$, is depicted in Figure 5. Note that, due to the shift from general to chosen pushouts, the source and target graphs of direct derivations are not preserved: It is enough

however to pre- and post-compose in the category $CoSpan(\mathbf{Graph})[K, \overline{D}]$ with suitable isomorphism cells, connecting the source G with the the chosen pushout G', as shown below in Definition 5.5.

Definition 5.5 (from rewrites to cells) *Let $\mathcal{G}$ be a disconnected* GTS *and let $\underline{\mathbf{G}}$ be the free bi-category generated by the computad $\mathbf{C}_{\mathcal{G}}$. Then, we denote by $\omega : D(\mathcal{G}) \to \underline{\mathbf{G}}$ the mapping that associates to each direct derivation $G \stackrel{p/d}{\Longrightarrow} H$ in $D(\mathcal{G})$ like in Figure A.1 the cell $\beta_{G,G'} \cdot ((\langle \emptyset, K, id_K \rangle * [p/d]) * \langle id_{\overline{D}}, \overline{D}, \emptyset \rangle) \cdot \beta_{H',H}$, for $\beta_{G,G'} : \langle \emptyset, G, \emptyset \rangle \Rightarrow \langle \emptyset, G', \emptyset \rangle$ the cell given by the unique arrow $G \to G'$ induced by the universal property, and similarly for $\beta_{H',H}$, and extends in the obvious way to many-step derivations.* □

The horizontal pre- and post-composition for $[p/d]$ is necessary in order to extend the map all derivations in the same hom-category, namely, $\underline{\mathbf{G}}[\emptyset, \emptyset]$. So, if there exists a direct derivation $G \stackrel{p/d}{\Longrightarrow} H$ in $D(\mathcal{G})$, then there exists a cell in the corresponding bi-category from $\langle \emptyset, G, \emptyset \rangle$ to $\langle \emptyset, H, \emptyset \rangle$.

It is easy to prove (see also [16]) that the inverse of the above statement holds. However, the characterisation of the mere existence of derivations by generated cells is not fully satisfactory. Instead, we would like to recover the concurrency properties of DPO graph rewriting as described in Section 3 by the parallelism theorem and the shift-equivalence. This is proved in the next theorem, our main result, stating that the mapping ω identifies two derivations if and only if they are shift-equivalent.

Theorem 5.6 (axiomatising shift-equivalence) *Let $\mathcal{G}$ be a disconnected* GTS*, and let ρ and σ be derivations in $D(\mathcal{G})$. Then, $\rho \equiv_{sh} \sigma$ iff $\omega(\rho) = \omega(\sigma)$.*

Proof (Sketch). The proof goes by structural induction, from cells to derivations; and by induction on the number of steps, from derivations to cells. Here, we just give the base case for the latter, when derivations are in direct as-correspondence. So, let us assume derivations $\rho = G \stackrel{p/d}{\Longrightarrow} H$ and $\sigma = G \stackrel{p_1/d_1}{\Longrightarrow} X \stackrel{p_2/d_2}{\Longrightarrow} H$ for productions $p_1, p_2 \in \mathcal{G}$. Let us first note that $[c_1, c_2] \circ \iota_i = c_i$, thus, by construction,

$$\begin{aligned}
[p/d] &= ([p_1/[c_1, c_2] \circ \iota_1] \otimes [p_2/[c_1, c_2] \circ \iota_2]) * \Delta_{\overline{D}} \\
&= ([p_1/c_1] \otimes [p_2/c_2]) * \Delta_{\overline{D}} \\
&= (([p_1/c_1] \otimes \langle c_{l_2} \circ l_2, G_2, g_2 \circ \delta_D \rangle) * \Delta_{\overline{D}}) \\
&\quad \cdot ((\langle c_{R_1} \circ r_1, H_1, h_1 \circ \delta_D \rangle \otimes [p_2/c_2]) * \Delta_{\overline{D}}),
\end{aligned}$$

by functoriality of $\otimes$ and $*$. Now, let us note that in the bi-category $\underline{\mathbf{G}}$, the cell family ? is strictly monoidal, namely, $?_{K_1 \otimes K_2} = ?_{K_1} \otimes ?_{K_2}$, so we have that

$$\begin{aligned}\omega(\rho) &= \beta_{G,G'} \cdot ((?_{K_1 \otimes K_2} * [p/d]) * !_{\overline{D}}) \cdot \beta_{H',H} \\ &= \beta_{G,G'} \cdot (((?_{K_1} \otimes ?_{K_2}) * (([p_1/c_1] \otimes [p_2/c_2]) * \Delta_{\overline{D}})) * !_{\overline{D}}) \cdot \beta_{H',H}.\end{aligned}$$

Let us restrict now our attention to the cell $\beta_{G,G'} \cdot ((?_{K_1} \otimes ?_{K_2}) * (([p_1/c_1] \otimes \langle c_{l_2} \circ l_2, G_2, g_2 \circ \delta_D \rangle) * \Delta_{\overline{D}}))$, and note that (denoting $\langle c_{l_1} \circ l_1, G_1, g_1 \circ \delta_D \rangle$ and $\langle c_{l_2} \circ l_2, G_2, g_2 \circ \delta_D \rangle$ as s_1 and s_2, respectively) the following arrows are all isomorphic, through cells induced by the monoidal structure of $\underline{\mathbf{G}}$.

$$\begin{aligned}\langle \emptyset, G', \emptyset \rangle &= ((?_{K_1} \otimes ?_{K_2}) * ((s_1 \otimes s_2) * \Delta_{\overline{D}})) * !_{\overline{D}} \\ &\cong (((?_{K_1} \otimes ?_{K_2}) * (s_1 \otimes s_2)) * \Delta_{\overline{D}}) * !_{\overline{D}} \\ &\cong ((?_{K_1} \otimes ?_{K_2}) * (s_1 \otimes s_2)) * (\Delta_{\overline{D}} * !_{\overline{D}}) \\ &\cong ((?_{K_1} * s_1) \otimes (?_{K_2} * s_2)) * (\Delta_{\overline{D}} * !_{\overline{D}}) \\ &\cong (((?_{K_1} * s_1) * id_{\overline{D}}) \otimes (id_e * (?_{K_2} * s_2)) * (\Delta_{\overline{D}} * !_{\overline{D}}) \\ &\cong (((?_{K_1} * s_1) \otimes id_e) * (id_{\overline{D}} \otimes (?_{K_2} * s_2)) * (\Delta_{\overline{D}} * !_{\overline{D}}) \\ &\cong ((?_{K_1} * s_1) \otimes id_e) * ((id_{\overline{D}} \otimes (?_{K_2} * s_2) * (\Delta_{\overline{D}} * !_{\overline{D}}))\end{aligned}$$

Now, please note than in our concrete category, $(id_{\overline{D}} \otimes (?_{K_2} * s_2)) * (\Delta_a * !_a)$ is isomorphic to the dual $\langle \emptyset, G_2, c_{l_2} \circ l_2 \rangle$, again *via* a cell induced by universality. [5] So, we obtain that $\beta_{G,G'} \cdot ((?_{K_1} \otimes ?_{K_2}) * (([p_1/c_1] \otimes \langle c_{l_2} \circ l_2, G_2, g_2 \circ \delta_D \rangle) * \Delta_{\overline{D}}))$ actually coincides with $\beta_{G,G''} \cdot \omega(G \overset{p_1/d_1}{\Longrightarrow} X) \cdot \beta_{X,X'}$. Similarly for the second half of the mapping of $\omega(\rho)$, and then the theorem holds. □

Together with the result of Theorem 3.6, this allows us to characterise the shift-equivalence on injective derivations by first disconnecting them and then applying Theorem 5.6 above.

Corollary 5.7 (axiomatising shift-equivalence) *Let $\mathcal{G}$ be an injective* GTS *and let ρ and σ be derivations in $D(\mathcal{G})$. Then, $\rho \equiv_{sh} \sigma$ iff $\omega(\delta(\rho)) = \omega(\delta(\sigma))$, where $\delta(\rho)$ and $\delta(\sigma)$ are the corresponding disconnected derivations of ρ and σ according to Proposition 3.5.* □

6 Future Work

This paper is part of an ongoing research effort, started with [16], in developing a categorical syntax for the DPO approach to graph rewriting. In the conclusions of [16] we pointed out that two main questions were left open. The first one involved the concurrency properties of graph derivations, since

[5] In fact, a duality property holds for dgs-monoidal categories, see [16].

"despite the correspondence we got is faithful at the level of the rewrite relation, we cannot recast the notions of *parallel derivation* and *shift equivalence* by means of the 2-categorical structure", due to the restriction we had there to discrete productions. The concurrency results for disconnected productions presented in Section 3 and Section 5 solved successfully this problem, since the equivalence induced on cells by the coherence axioms of bi-categories coincides with shift equivalence on graph derivations [6], like it happens for *permutation equivalence* in categorical models of term rewriting (see e.g. [9]).

The second problem concerning the rewriting on abstract graphs was avoided (rather than solved) by the use of bi-categories. In fact, our original goal, an axiomatic description of *abstract* graph rewriting, is not yet fully achieved. But we believe that the coherence theorem for bi-categories (stating that every bi-category is bi-equivalent to a 2-category) may provide the solution to this problem. In fact, with our interpretation, this bi-equivalence represents the transition from concrete to abstract states while preserving for any two given states the number of derivations between them (and thus the amount of concurrency). Technically speaking, the coherence isomorphisms of bi-categories provide us with an algebraic description of the standard isomorphisms in [11].

The final step, then, consists in replacing the concrete bi-category $CoSpan^{\oplus,iso}(\mathbf{Graph})$ by a syntactic (e.g., freely generated) bi-category, eventually equipped with a dgs-monoidal structure. Such a construction has been given already for the category of abstract spans in **Graph** with discrete interfaces in [16], and we think that it can be straightforwardly extended to disconnected graphs by adding appropriate generators for the edges. Notice that, in contrast to [16], dgs-monoidal categories in this paper are not required to be *symmetric*: While this additional structure is essential for the generation of graphs, it is not needed for axiomatising shift-equivalence on derivations. In fact, all graph derivations are represented by cells of the hom-category on the empty interface graph (the unit object), and it could be shown that this hom-category is symmetric monoidal. The reader would have noticed that, in fact, most of the additional structure on bi-categories we introduced is redundant for our purposes, namely, the proof of Theorem 5.6: We left it just to suggest the forthcoming direction of our work, and to emphasise the relationship with the solution we previously proposed, and with related categorical approaches.

References

[1] M. Bauderon and B. Courcelle. Graph expressions and graph rewritings. *Mathematical Systems Theory*, 20:83–127, 1987.

[2] J. Bénabou. Introduction to bicategories. In *Midwest Category Seminar*, volume 47 of *Lectures Notes in Mathematics*, pages 1–77. Springer Verlag, 1967.

[3] D.B. Benson. The basic algebraic structures in categories of derivations.

Information and Control, 28:1–29, 1975.

[4] A. Carboni, S. Kasangian, and R.H. Street. Bicategories of spans and relations. *Journal of Pure and Applied Algebra*, 33:259–267, 1984.

[5] A. Carboni and R.F.C. Walters. Cartesian bicategories I. *Journal of Pure and Applied Algebra*, 49:11–32, 1987.

[6] A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi. Abstract graph derivations in the double-pushout approach. In H.-J. Schneider and H. Ehrig, editors, *Graph Transformations in Computer Science*, volume 776 of *LNCS*, pages 86–103. Springer Verlag, 1994.

[7] A. Corradini and F. Gadducci. A 2-categorical presentation of term graph rewriting. In E. Moggi and G. Rosolini, editors, *Category Theory and Computer Science*, volume 1290 of *LNCS*, pages 87–105. Springer Verlag, 1997.

[8] A. Corradini and F. Gadducci. Rewriting on cyclic structures: Equivalence between the operational and the categorical description. *Informatique Théorique et Applications/Theoretical Informatics and Applications*, 1999. To appear.

[9] A. Corradini, F. Gadducci, and U. Montanari. Relating two categorical models of term rewriting. In J. Hsiang, editor, *Rewriting Techniques and Applications*, volume 914 of *LNCS*, pages 225–240. Springer Verlag, 1995.

[10] A. Corradini and U. Montanari. An algebraic semantics for structured transition systems and its application to logic programs. *Theoret. Comput. Sci.*, 103:51–106, 1992.

[11] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation I: Basic concepts and double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation*. World Scientific, 1997.

[12] B. Day and R.H. Street. Monoidal bicategories and Hopf algebroids. *Advances in Mathematics*, 129:99–157, 1997.

[13] H. Ehrig. Introduction to the algebraic theory of graph grammars. In V. Claus, H. Ehrig, and G. Rozenberg, editors, *Graph-Grammars and Their Application to Computer Science and Biology*, volume 73 of *LNCS*, pages 1–69. Springer Verlag, 1979.

[14] H. Ehrig and H.-J. Kreowski. Categorical approach to graphic systems and graph grammars. In *Algebraic System Theory*, volume 131 of *Lecture Notes in Economics and Mathematical Systems*, pages 323–351. Springer Verlag, 1975.

[15] H. Ehrig, M. Pfender, and H.J. Schneider. Graph-grammars: an algebraic approach. In R.V. Book, editor, *Switching and Automata Theory*, pages 167–180. IEEE Computer Society Press, 1973.

[16] F. Gadducci and R. Heckel. An inductive view of graph transformation. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS*, pages 219–233. Springer Verlag, 1998.

[17] R. Gordon, A.J. Power, and R.H. Street. *Coherence for Tricategories*, volume 158 of *Memoirs of the American Mathematical Society*. American Mathematical Society, 1995.

[18] M. Hasegawa. *Models of Sharing Graphs*. PhD thesis, University of Edinburgh, 1997.

[19] R. Heckel. *Open Graph Transformation Systems: A New Approach to the Compositional Modelling of Concurrent and Reactive Systems*. PhD thesis, Technische Universität Berlins, 1998.

[20] P. Katis, N. Sabadini, and R.F.C. Walters. Bicategories of processes. *Journal of Pure and Applied Algebra*, 115:141–178, 1997.

[21] P. Katis, N. Sabadini, and R.F.C. Walters. SPAN(Graph): A categorical algebra of transition systems. In M. Johnson, editor, *Algebraic Methodology and Software Technology*, volume 1349 of *LNCS*, pages 307–321. Springer Verlag, 1997.

[22] G.M. Kelly and R.H. Street. Review of the elements of 2-categories. In G.M. Kelly, editor, *Sydney Category Seminar*, volume 420 of *Lecture Notes in Mathematics*, pages 75–103. Springer Verlag, 1974.

[23] H.-J. Kreowski. *Manipulation von Graphmanipulationen*. PhD thesis, Technische Universität Berlin, 1977.

[24] T. Leinster. Basic bicategories. Available at `http://can.dpmms.cam.ac.uk/~leinster/`, 1998.

[25] J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoret. Comput. Sci.*, 96:73–155, 1992.

[26] J. Meseguer and U. Montanari. Petri nets are monoids. *Information and Computation*, 88:105–155, 1990.

[27] A.J. Power. An abstract formulation for rewrite systems. In D.H. Pitt, D.E. Rydehard, P. Dybjer, A.M. Pitts, and A. Poigné, editors, *Category Theory and Computer Science*, volume 389 of *LNCS*, pages 300–312. Springer Verlag, 1989.

[28] G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation*. World Scientific, 1997.

[29] D.E. Rydehard and E.G. Stell. Foundations of equational deductions: A categorical treatment of equational proofs and unification algorithms. In D.H. Pitt, A. Poigné, and D.E. Rydehard, editors, *Category Theory and Computer Science*, volume 283 of *LNCS*, pages 114–139. Springer Verlag, 1987.

[30] J.G. Stell. *Categorical Aspects of Unification and Rewriting*. PhD thesis, University of Manchester, 1992.

[31] R.H. Street. Categorical structures. In M. Hazewinkel, editor, *Handbook of Algebra*, pages 529–577. Elsevier, 1996.

A An introductory appendix

The aim of this appendix is to provide the reader with the precise definitions on both graph transformations and bi-categories that have been used throughout the paper. Their (rather) standard nature convinced us to include them in an appendix, in order not to slow down the knowledgeable reader, yet allowing for the article to be self-contained.

A.1 On the DPO approach to graph transformation

Our presentation of graph transformation is tailored over the needs of our representation theorems of Section 3 and Section 5.

Definition A.1 (graph) *A* (directed) graph *d is a four-tuple $\langle E, N, s, t\rangle$ such that E is the set of* edges*, N is the set of* nodes*, and $s, t : E \to N$ are the* source *and* target *functions.* □

Given a graph $d = \langle E, N, s, t\rangle$, its components shall often be denoted by N_d, E_d, s_d and t_d, respectively.

Definition A.2 (graph morphism) *Let d and d' be graphs. A* graph morphism *$f : d \to d'$ is a pair of functions $\langle f_e, f_n\rangle$ such that $f_e : E_d \to E_{d'}$ and $f_n : N_d \to N_{d'}$. These functions must preserve source and target, i.e., for each edge $e \in E_d$, $s_{d'}(f_e(e)) = f_n(s_d(e))$, and $t_{d'}(f_e(e)) = f_n(t_d(e))$. Graphs and graph morphisms form a category denoted by* **Graph**. □

In the literature on graph rewriting, it is almost customary to consider only *injective* productions, as e.g. in the survey article [11]. This ensures interesting concurrency properties, like the existence of canonical representatives of equivalence classes of derivations with respect to *shift*-equivalence. In this paper, non-injective productions arise when *disconnecting* the interface of an injective production (cf. Definition 3.3). Hence, in the following definitions, no assumptions are made about the injectivity of production morphisms.

Definition A.3 (graph production and derivation) *A* graph production *$p : s$ is composed of a* production name *p and of a span of (not necessarily injective) graph morphisms $s = (L \stackrel{l}{\longleftarrow} K \stackrel{r}{\longrightarrow} R)$. A* graph transformation system *(or* GTS*) $\mathcal{G}$ is a set of productions, all with different names. Thus, when appropriate, we denote a production $p : s$ using only its name p.*

A graph production $p : (L \stackrel{l}{\longleftarrow} K \stackrel{r}{\longrightarrow} R)$ is injective *if both l and r are injective. A graph transformation system $\mathcal{G}$ is* injective *if all its productions are so.*

A double-pushout *diagram is a diagram like in Figure A.1, where top and bottom are spans and (1) and (2) are pushouts squares in* **Graph**. *If $p : (L \stackrel{l}{\longleftarrow}$*

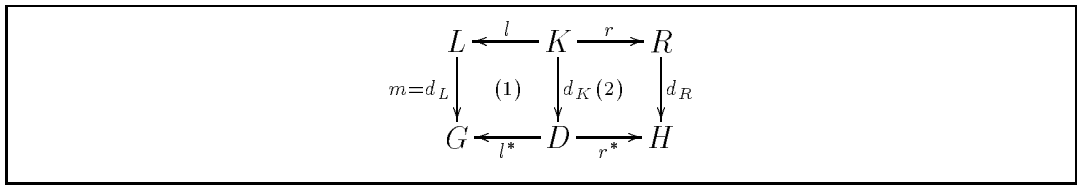

Fig. A.1: A DPO direct derivation

$K \xrightarrow{r} R$) *is a production, a* direct derivation *from* G *to* H *via production* p *and* $d = \langle d_L, d_K, d_R \rangle$ *is denoted by* $G \overset{p/d}{\Longrightarrow} H$.

A derivation *in a* GTS $\mathcal{G}$ *is a finite sequence of direct derivations* $G_0 \overset{p_1/d_1}{\Longrightarrow} \ldots \overset{p_n/d_n}{\Longrightarrow} G_n$ *where* $p_1, \ldots, p_n$ *are productions of* $\mathcal{G}$. □

Operationally, the application of a production p to a graph G consists of three steps. First, the *match* $m : L \to G$ is chosen, providing an occurrence of L in G. Then, all objects of G matched by $L - l(K)$ are removed, leading to the context graph D. Finally, the objects of $R - r(K)$ are added to D, obtaining the derived graph H.

The construction of context graph D can be more abstractly described as a *pushout complement*, which is formally given by the graph D itself and morphisms l^* and d_K such that the square in the left of Figure A.1 is a pushout. The existence of the pushout complement (1), and hence of a direct derivation[6] $G \overset{p/d}{\Longrightarrow} H$, is characterised by the *gluing conditions* [13]: The *dangling condition* ensures that the structure D obtained by removing from G all objects to be deleted is indeed a graph, that is, no edges are left "dangling" without source or target node. The *identification condition* states that objects from the left-hand side may only be identified by the match if they also belong to the interface (and are thus preserved).

A.2 On monoidal bi-categories

Roughly, a *bi-category* $\underline{\mathbf{C}}$ can be presented as a set of objects, $Ob_{\mathbf{C}}$, and, for each pair of objects, a category $\underline{\mathbf{C}}[a, b]$. For the sake of space, in the following we present the axioms equationally, instead of using the more intuitive diagrammatic way as in [24].

Definition A.4 (bi-categories) *A* bi-category *is a seven-tuple* $\langle Ob_{\mathbf{C}}, \underline{\mathbf{C}}, *, id, \alpha, \rho, \lambda \rangle$ *such that* $Ob_{\mathbf{C}}$ *is a set of* objects *and, indexed by elements in* $Ob_{\mathbf{C}}$, $\underline{\mathbf{C}}$ *is a family of categories* $\underline{\mathbf{C}}[a, b]$ *(the* hom-categories *of* $\underline{\mathbf{C}}$*),* $*$ *is a family of functors* $*^b_{a,c} : \underline{\mathbf{C}}[a, b] \times \underline{\mathbf{C}}[b, c] \to \underline{\mathbf{C}}[a, c]$, id *is a family of objects* $id_a \in |\underline{\mathbf{C}}[a, a]|$, $\alpha : f * (g * h) \Rightarrow (f * g) * h$, $\lambda : f * id_a \Rightarrow f$ *and* $\rho : id_a * f \Rightarrow f$ *are natural isomorphisms, satisfying the axioms*[7]

[6] The pushout (2) always exists since **Graph** is co-complete.

[7] For the sake of readability, all the indexes are dropped, either of $*$, or of the composition $\cdot$ inside the hom-categories. Moreover, the identity of an object of the hom-categories, such as f or id_a, is usually denoted by the object itself.

- $\alpha_{f,id_a,g} \cdot (\lambda_f * g) = f * \rho_g$, *and*
- $(f * \alpha_{g,h,k}) \cdot \alpha_{f,g*h,k} \cdot (\alpha_{f,g,h} * k) = \alpha_{f,g,h*k} \cdot \alpha_{f*g,h,k}$. □

We denote as *arrows* and *cells* of a bi-category $\underline{\mathbf{C}}$ the objects and arrows of the hom-categories, respectively, and by $\beta : f \Rightarrow g : a \to b$ we mean that β is a cell in $\underline{\mathbf{C}}[a,b]$ from f to g. Since each $\underline{\mathbf{C}}[a,b]$ is a category, cells can be composed *vertically*, that is, if $\gamma : g \Rightarrow h : a \to b$ is another cell we can form the composite $\beta \cdot \gamma : f \Rightarrow h : a \to b$. Moreover, there is an operation of *horizontal composition* of cells $\beta : f \Rightarrow g : a \to b$ and $\beta' : f' \Rightarrow g' : b \to c$ to $\beta * \beta' : f' * f \Rightarrow g' * g : a \to c$. In computational models based on 2-categories (like [7,27]), the cells represent computations of the system which can be composed sequentially and in parallel using, respectively, the operations of vertical and horizontal composition of the 2-category: We will see in Section 5 in which sense a similar analogy holds also for our bi-categorical setting.

Definition A.5 (morphisms, bi-transformations and modifications) *Let* $\underline{\mathbf{C}}$, $\underline{\mathbf{D}}$ *be bi-categories. A morphism* $(F,\phi) : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$ *consists of a function* $F : Ob_{\mathbf{C}} \to Ob_{\mathbf{D}}$, *a family of functors* $F_{a,b} : \underline{\mathbf{C}}[a,b] \to \underline{\mathbf{D}}[F(a),F(b)]$ *and natural transformations* $\phi_{f,g} : F(f) * F(g) \Rightarrow F(f*g)$ *and* $\phi_a : id_{F(a)} \Rightarrow F(id_a)$ *satisfying the axioms*

- $(F(f) * \phi_{g,h}) \cdot \phi_{f,g*h} \cdot F(\alpha_{f,g,h}) = \alpha_{F(f),F(g),F(h)} \cdot (\phi_{f,g} * F(h)) \cdot \phi_{f*g,h}$,
- $(\phi_a * F(f)) \cdot \phi_{id_a,f} \cdot F(\rho) = \rho_{F(f)}$, *and*
- $(F(f) * \phi_b) \cdot \phi_{f,id_b} \cdot F(\lambda) = \lambda_{F(f)}$.

A homomorphism (F,ϕ) *is a morphism such that the components of* ϕ *are isomorphisms; it is* strict *if the components are identities.*

A bi-transformation $\sigma : (F,\phi) \Rightarrow (G,\xi)$ *between morphisms consists of a family of arrows* $\sigma_a : F(a) \to G(a)$ *and natural transformations* $\sigma_f : \sigma_a * G(f) \Rightarrow F(f) * \sigma_b$ *satisfying the axioms*

- $\lambda_{\sigma_a} \cdot \rho^{-1}_{\sigma_a} \cdot (\phi_a * \sigma_a) = (\sigma_a * \xi_a) \cdot \sigma_{id_a}$, *and*
- $\alpha_{\sigma_a,G(f),G(g)} \cdot (\sigma_f * G(g)) \cdot a^{-1}_{F(f),\sigma_b,G(g)} \cdot (F(f) * \sigma_g) \cdot \alpha_{F(f),F(g),\sigma_c} \cdot (\phi_f * \sigma_c)$
  $= (\sigma_a * \xi_{f,g}) \cdot \sigma_{f*g}$.

A bi-transformation is strong *if the components of* σ *are isomorphisms.*

A modification $\Gamma : \sigma \Rrightarrow \pi : (F,\phi) \Rightarrow (G,\xi)$ *between bi-transformations consists of a family of cells* $\Gamma_a : \sigma_a \Rightarrow \pi_a : F(a) \to G(a)$ *satisfying the axiom*

- $(\Gamma_a * G_f) \cdot \pi_f = \sigma_f \cdot (F_f * \Gamma_b)$.

A modification is invertible *if the components of* Γ *are isomorphisms.* □

We have now all the machinery to introduce *monoidal* bi-categories [17].

Definition A.6 (monoidal bi-categories) *A* monoidal *bi-category is a eight-tuple* $\langle \underline{\mathbf{C}}, \otimes, I, \alpha', \rho', \lambda', \Pi, \Xi \rangle$ *such that* $\underline{\mathbf{C}}$ *is a bi-category;* $\otimes : \underline{\mathbf{C}} \times \underline{\mathbf{C}} \to \underline{\mathbf{C}}$ *and* $e : \underline{\mathbf{1}} \Rightarrow \underline{\mathbf{C}}$ *are homomorphisms;*[8] *'*$\alpha' : a \otimes (b \otimes c) \Rightarrow (a \otimes b) \otimes c$, $\lambda' : a \otimes e \Rightarrow a$

[8] For $\underline{\mathbf{1}}$ the unit bi-category: We denote by e also the corresponding object in $\underline{\mathbf{C}}$.

and $\rho' : e \otimes a \Rightarrow a$ *are strong bi-transformations; and*

- $\Xi : \alpha'_{a,e,b} * (\lambda'_a \otimes b) \Rrightarrow a \otimes \rho'_b$, *and*
- $\Pi : (a \otimes \alpha'_{b,c,d}) * \alpha'_{a,b\otimes c,d} * (\alpha'_{a,b,c} \otimes d) \Rrightarrow \alpha'_{a,b,c\otimes d} * \alpha'_{a\otimes b,c,d}$,

are invertible modifications satisfying (the instances of) the non-abelian 4-cocycle condition *and the* left *and* right normalisation *axioms in Definition 2.2. of [17].*

Let $\underline{\mathbf{C}}$, $\underline{\mathbf{D}}$ *be monoidal bi-categories. A* monoidal *morphism* $(F, \phi, \eta) : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$ *consists of a morphism* $(F, \phi) : \underline{\mathbf{C}} \to \underline{\mathbf{D}}$*; strong bi-transformations* $\eta_{a,b} : F(a) \otimes F(b) \Rightarrow F(a \otimes b)$ *and* $\eta_e : e \Rightarrow F(e)$*; and invertible modifications*

- $(F(a) \otimes \eta_{b,c}) * \eta_{a,b\otimes c} * F(\alpha'_{a,b,c}) \Rrightarrow \alpha'_{F(a),F(b),F(c)} * (\eta_{a,b} \otimes F(c)) * \eta_{a\otimes b,c}$,
- $(\eta_e \otimes F(a)) * \eta_{id_e,a} * F(\rho') \Rrightarrow \rho'_{F(a)}$, *and*
- $(F(a) * \eta_e) * \eta_{a,id_e} * F(\lambda') \Rrightarrow \lambda'_{F(a)}$

satisfying (the instances of) the axioms in Definition 3.1 of [17].

We apologise to the reader for spelling out the previous definition in an incomplete way. We just wanted to give the reader a chance to check the few diagrams we used, thus explicitly mentioning the modifications involved, without dealing with coherence issues, and simply assuming the problem as solved.